A survey on hard subspace clustering algorithms

[1]A. Surekha, [2]S. Anuradha, [3]B. Jaya Lakshmi, [4]K. B. Madhuri

[1,2]Research Scholars, [3]Assistant Professor, [4]HOD
Gayatri Vidya Parishad College of Engineering (Autonomous), Visakhapatnam, India

***Abstract*---Subspace clustering is an extension to traditional clustering that seeks to find clusters in different subspaces within a dataset. Subspace clustering finds sets of objects that are homogeneous in subspaces of high-dimensional datasets, and has been successfully applied in many domains. Often in high dimensional data, many dimensions may be irrelevant and can mask real clusters. Subspace clustering algorithms localize the search process for relevant dimensions allowing them to find clusters that exist in various subspaces. Subspace clustering can be categorized into hard subspace clustering (HSC) and soft subspace clustering (SSC). HSC algorithms assume that each dimension in the data set has equal importance in the process of clustering, while SSC algorithms deal with feature weighing based on its contribution. Based on the direction of exploration of subspace clusters, HSC algorithms could be classified into two main categories: Top-down and Bottom-up. Top-down algorithms find an initial clustering in the full set of dimensions and evaluate the subspaces of each cluster, iteratively improving the results. Bottom-up approaches find dense regions in low dimensional spaces and combine them to form clusters. This paper surveys various hard subspace clustering algorithms and their efficacies, insufficiencies and recent developments. The readers would be provided with clear outline about the existing algorithms and nurture further developments and significant research in the area.**

***Index Terms*---Subspace Clustering, Hard Subspace Clustering, Top-down approach, Bottom-up approach.**

I. Introduction

One of the primary data mining task is Clustering. Most real-world datasets are characterized by a high-dimensional sparse data space where meaningful clusters may not be detected in traditional clustering algorithms [11]. However, the datasets often contain attractive clusters which are concealed in various subspaces of the original feature space. Many applications such as molecular biology, geography, finance and marketing produce enormous amounts of data which cannot be managed no longer without the help of efficient and effective data mining methods. Subspace Clustering finds sets of objects that are homogeneous in subspaces of high-dimensional datasets and has been successfully applied in many applications.

Many clustering algorithms face the problem of curse of dimensionality when high-dimensional data is used. The distance measures become insignificant gradually, as the number of dimensions increases in a dataset. Additional dimensions spread out the points until, in very high dimensions; they are almost equidistant from each other.

Some of the techniques which deal with the problem of curse of dimensionality are both feature transformation and feature selection techniques [11]. Feature transformation techniques attempt to summarize a dataset in fewer dimensions by creating combinations of the original attributes. However, since they preserve the relative distances between objects, they are less effective when there are large numbers of irrelevant attributes that hide the clusters in sea of noise. Also, the new features are combinations of the originals and may be very difficult to interpret the new features in the context of the domain. Feature selection methods select only the most relevant of the dimensions from a dataset to reveal groups of objects that are similar on only a subset of their attributes. While quite successful on many datasets, feature selection algorithms have difficulty when clusters are found in different subspaces [11].

Subspace clustering is an extension of feature selection which tries to identify clusters in different subspaces of the same dataset. Like feature selection, subspace clustering needs a search method and an evaluation criteria. In addition, subspace clustering must somehow restrict the scope of the evaluation criteria so as to consider different subspaces for each different cluster [15].

The problem of subspace clustering is often divided into two sub problems. They are determining the subspaces and the clustering data. Based on how these problems are addressed there are two main categories of subspace clustering methods. They are Hard Subspace Clustering (HSC) and Soft Subspace Clustering (SSC). SSC algorithms perform clustering in high-dimensional spaces by assigning a weight to each dimension to measure the contribution of individual dimensions to the formation of a particular cluster [15]. In Hard Subspace Clustering a feature in a subspace contributes equally in the clustering process.

II. CLASSIFICATION OF HARD SUBSPACE CLUSTERING ALGORITHMS

Based on the approaches of how these subspace clusters are discovered the hard subspace clustering algorithms are divided into two main categories. They are Bottom-up and Top-Down approaches.

In Bottom-up approach the subspace clusters are discovered from lower dimensional space to higher dimensional space. The bottom-up search method makes use of the downward closure property of density to reduce the search space, using an APRIORI property. Algorithms first create a histogram for each dimension and select those bins with densities above a given threshold. The downward closure property of density meant that if there are dense units in k dimensions, there are dense units in all (k -1) dimensional projections [13]. Candidate subspaces in two dimensions can then be formed using only those dimensions which contained dense units, dramatically reducing the search space.

Bottom-up Algorithms:

- CLIQUE
- OPTIGRID
- MAFIA
- SUBCLU
- FIRES
- DENCOS

The top-down subspace clustering approach begins by finding an initial approximation of the clusters in the full dimensional space with uniform weighted dimensions. Next each dimension is assigned a weight for each cluster. The updated weights are made used in the next iteration to regenerate the clusters. This approach needs many iterations of clustering algorithms in the full set of dimensions. Many of the implementations of this strategy make use of sampling technique to improve performance. Top-down algorithms create clusters that are partitions of the dataset, meaning each instance is assigned to only one cluster [13]. Many algorithms also allow for an additional group of outliers. Parameter tuning is necessary in order to get meaningful results.

Top-Down Algorithms:

- PROCLUS
- ORCLUS
- FIND-IT
- δ-CLUSTERS
- COSA

III. BOTTOM-UP ALGORITHMS

CLustering In QUEst (CLIQUE)

CLIQUE deals with the problems ensured in clustering high dimensional data. By taking grid size and a density threshold values as a user input, the algorithm CLIQUE find clusters of arbitrary shape in large dimensional data. The process starts by finding clusters at a single dimension and then proceeds towards high dimensions.

The algorithm CLIQUE is a bottom-up subspace clustering algorithm that constructs static grids. To reduce the search space the clustering algorithm uses apriori approach. CLIQUE is both grid- based and density based subspace clustering algorithm [12]. To find out the clusters, density threshold and number of grids are taken as input parameters. CLIQUE operates on multidimensional data. It does not operate all the dimensions at once but by processing a single dimension at first step and then grows upward to the higher-one. According to given the grid size, the clustering process in CLIQUE involves first dividing the number of dimensions into non- overlapping rectangular units called grids and find out the dense region according to a given threshold value. A unit is dense if the data points in this are exceeding the threshold value. By using the apriori approach the clusters are generated from all dense subspaces. Finally CLIQUE algorithm generates minimal description for the clusters obtained by first determines the maximal dense regions in the subspaces and then minimal cover for each cluster from that maximal region. It repeats the same procedure until all the dimensions are covered.

OPTImal GRID clustering (OPTIGRID)

OptiGrid is a clustering technique which is based on constructing an optimal grid-partitioning of the data. The optimal grid-partitioning is determined by calculating the best partitioning hyper planes for each dimension (if such a partitioning exists) using certain projections of the data. The algorithm works recursively. In each step, it partitions the actual data set into a number of subsets if possible. The subsets which contain at least one cluster are treated recursively [1]. The partitioning is done using a multidimensional grid defined by at most q cutting planes. Each cutting plane is orthogonal to at least one projection. The point density at cutting planes is bound by the density of the orthogonal projection of the cutting plane in the projected space. The q cutting planes are chosen to have a minimal point density. The recursion stops for a subset if no good cutting plane can be found any more. There are two desirable properties for good cutting planes: First, cutting planes should partition the data set in a region of low density (the density should be at least low relative to the surrounding region) and second, a cutting plane should discriminate clusters as much as possible. The first constraint guarantees that a cutting plane does not split a cluster, and the second constraint makes sure that the cutting plane contributed to finding the clusters. Without the second constraint, cutting planes are best placed at the borders of the data space because of the minimal density there, but it is obvious that in that way clusters cannot be detected.

Merging of Adaptive Finite Intervals (MAFIA)

MAFIA introduces the use of adaptive grids for efficient and scalable computation of clusters in subspaces of large data sets and large number of dimensions. To compute the dense units in all dimensions of bottom-up algorithm for subspace clustering and combines these to generate the dense units in higher dimensions [5].

An adaptive interval size is proposed to partition the dimension based on the distribution of data in the dimension. The minimum number of bins for a dimension has been determined, by one pass of the data initially using the construction of histogram. Contiguous bins with similar histogram values are combined to form larger bins. To reduce the computation, the bins and cells which have low density of data will be pruned limiting the eligible candidate dense units. As in the uniform bins case, the bin boundaries will also not be rigid and the cluster boundaries will be able to delineate in each dimension. The clustering result quality will be improved. To reflect the window value, the maximum value of the histogram will be set firstly, in a dimension within a small window. Adjacent windows are merged to form larger windows within a certain threshold [5].

Density-connected SUBspace CLUstering (SUBCLU)

SUBCLU is an efficient and effective approach to the subspace clustering problem. The concept of density-connectivity is used which underlies the DBSCAN algorithm. The number of objects that lie around an object within a certain radius is called the neighborhood of that object. If the number of objects in the neighborhood exceeds minpts a minimum number of points that object is called core object. An object p is directly density-reachable from q if p is in the neighborhood of q and q is a core object. An object p isdensity-reachable from q if there exists a series of points x1…….xn such that x1=q and xn=p and xi+1is directly-density reachable from xi for all i=1to n. An object p is density-connected to q if there exists an object o such that both p and q are density-reachable from o. SUBCLU searches for objects which are density-connected. Firstly, it generates all the subspace clusters in the lower-dimensions. Then it keeps on pruning the subspaces based on apriori property [9]. This process continues all the subspace clusters in higher-dimensions are discovered. In contrast to existing grid-based approaches, SUBCLU is able to detect arbitrarily shaped and positioned clusters in subspaces. In the process of generating all clusters in a bottom-up way, the monotonicity of density-connectivity is used efficiently to prune subspaces [9]. While not examining any unnecessary subspaces, SUBCLU delivers for each subspace the same clusters DBSCAN would have found, when applied to this subspace separately.

FIlter REfinement Subspace clustering (FIRES)

Filter Refinement Subspace clustering (FIRES) is a general framework for efficient subspace clustering. It is generic in such a way that it works with all kinds of clustering notions. It starts with 1D clusters that can be constructed with a clustering method of choice and merges these 1D clusters to generate approximations of subspace clusters. An optional refinement step can compute the true subspace clusters, again using any clustering notion of choice and a variable cluster criterion [6]. FIRES consists of the following three steps:

Firstly, in Pre-clustering step all 1D clusters called base clusters are computed. This is similar to existing subspace clustering approaches and can be done using any clustering algorithm of choice. For pre-clustering, the filter step which drops irrelevant base-clusters, that do not contain any vital subspace cluster information. Small base-clusters do not likely include significant subspace clusters, because they usually indicate a sparse area in the higher dimensional space.

Secondly, in generation of subspace cluster approximations step, the base-clusters are merged to find maximal-dimensional subspace cluster approximations. Approximations of maximal-dimensional subspace clusters are determined by suitably merging the base-clusters derived from the preprocessing step. Out of all merge possibilities, whose number is exponential in the number of base-clusters, the most promising merge candidates are to be found by searching for all base-clusters which contain nearly the same objects.

Consequently, a good indication for the existence of a subspace cluster is if the intersection between base-clusters is large. The base-clusters are similar if they share a sufficiently high number of objects [6]. More generally, each base-cluster which includes more than one overlapping subspace cluster should be split into multiple base-clusters in such a way, that each of them contains at most one of the subspace clusters.

Thirdly, in post processing of subspace clusters step, significant subspace clusters should achieve a good trade-off between dimensionality and cluster size. Therefore, post-processing steps like pruning and refinement are required in order to achieve good results. Pruning improves the quality of the merge able-cluster-set by identifying and removing “meaningless” base-clusters. Refinement removes noise and completes the subspace clusters. There may be clusters in the subspace cluster approximations which do not contain relevant information of the corresponding subspace cluster. Two building methods can be distinguished, the union and the intersection. Obviously, both variants do not yield the true subspace cluster. Due to the fact that the base-clusters contain a lot of noise in a higher dimensional subspace, the intersection variant seems to produce more accurate approximations than the union variant. However the intersection merge has significant draw-backs because the detected subspace cluster would be too strict, thus many promising candidates would be lost. Furthermore, parts of the true subspace cluster can be outside of the approximation which would lead to incomplete results. We achieve better subspace cluster results when applying an additional refinement step. Firstly, the union of all base-clusters is computed in order to avoid that potential cluster. Then, the merged set of objects in the corresponding subspace is clustered again. Thereby, any clustering algorithm can be applied, e.g. DBSCAN.

Density Conscious Subspace Clustering (DENCOS):

In many density- based approaches clusters in a subspace are regarded as regions of high- density and that are separated by lower- density regions. These approaches ignore the fact that the cluster densities vary in different subspace cardinalities. This problem of “density divergence” is dealt in this technique. The density thresholds for clusters vary due to the requirement of different dimensions, it is challenging in subspace clustering to simultaneously achieve high precision and recall for clusters in different subspace cardinalities [14]. To extract clusters with different density thresholds in different cardinalities is useful but is quite challenging. To efficiently discover dense units, a practicable way would be to store the complete information of the dense units in all subspace cardinalities into a compact structure such that the mining process can be directly performed in memory without repeated database scans. The idea is motivated to construct a compact structure which is extended from the FP-tree to store the crucial information of the data set. The base idea is by transforming the problem of identifying "dense units" in subspace clustering into a similar problem of discovering "frequent itemsets" in association rule mining. Thus the compact structure is constructed by storing the complete information of the dense units and the different thresholds in different subspace cardinalities is satisfied by the dense units and the dense units can be discovered from this structure efficiently.

IV. TOP-DOWN ALGORITHMS

PROjected CLUstering (PROCLUS)

In high dimensional spaces not all dimensions may be relevant to a given cluster. One way of handling this is to pick the closely correlated dimensions and find clusters in the corresponding subspace. Therefore by generalizing the clustering problem, referred to as the projected clustering problem, in which the subsets of dimensions selected are specific to the clusters themselves. The problem of finding projected clusters is two-fold: the cluster centers must be located and the appropriate set of dimensions is to be determined in which each cluster exists. A well known general approach is the so-called K Medoids method, which uses points in the original data set to serve as surrogate centers for clusters during their creation. Such points are referred to as medoids. One method which uses the K-Medoids approach, called CLARANS, for clustering in full dimensional space [2].

The algorithm proceeds in three phases: an initialization phase, an iterative phase, and a cluster refinement phase. The general approach is to find the best set of medoids by a hill climbing process similar to the one used in CLARANS, but generalized to deal with projected clustering. “Hill climbing” is the process of successively improving a set of medoids, which serve as the anchor points for the different clusters [3]. The purpose of the initialization phase is to reduce the set of points on which the hill climbing is performed, while at the same time trying to select representative points from each cluster in this set. The second phase represents the hill climbing process that is used in order to find a good set of medoids. A set of dimensions are also computed corresponding to each medoid so that the points assigned to the medoid best form a cluster in the subspace determined by those dimensions. The assignment of points to medoids is based on Manhattan segmental distances relative to these sets of dimensions. Thus, the search is performed not just in the space of possible medoids but also in the space of possible dimensions associated with each medoid. Finally, a cluster refinement phase is performed in which one pass over the data is used in order to improve the quality of the clustering.

arbitrarily ORiented projected CLUSter generation (ORCLUS)

High dimensional data has always been a challenge for clustering algorithms because of the inherent sparsity of the points. The subspaces are specific to the clusters themselves. This definition is substantially more general and realistic than currently available techniques which limit the method to only projections from the original set of attributes [4]. The generalized projected clustering technique may also be viewed as a way of trying to redefine clustering for high dimensional applications by searching for hidden subspaces with clusters which are created by inter-attribute correlations. In order to make the algorithm scalable for very large databases an extended cluster feature vectors is used.

A Fast and Intelligent Subspace Clustering Algorithm using Dimension Voting (FIND-IT):

In subspace clustering, selecting correct dimensions is very important because the distance between points is easily changed according to the selected dimensions. However, to select dimensions correctly is difficult, because data grouping and dimension selecting should be performed simultaneously. FINDIT determines the correlated dimensions for each cluster based on two key ideas: dimension-oriented distance measure which fully utilizes dimensional difference information, and dimension voting policy which determines important dimensions in a probabilistic way based on V nearest neighbors’ information [10].

A Fast and Intelligent Subspace Clustering Algorithm using Dimension Voting, FINDIT is similar in structure to PROCLUS and the other top-down methods, but uses a unique distance measure called the Dimension Oriented Distance (DOD). The idea is compared to voting whereby the algorithm tallies the number of dimensions on which two instances are within a threshold distance, ε of each other. The concept is based on the assumption that in higher dimensions it is more meaningful for two instances to be close in several dimensions rather than in a few. The algorithm typically consists of three phases, namely sampling phase, cluster forming phase and data assignment phase. The algorithms starts by selecting two small sets generated through random sampling of the data. The sets are used to determine initial representative medoids of the clusters. In the cluster forming phase the correlated dimensions are found using the DOD measure for each medoid. FINDIT then increments the value of ε and repeats this step until the cluster quality stabilizes. In the final phase, all of the instances are assigned to medoids based on the subspaces found.

FINDIT requires two input parameters, the minimum number of instances in a cluster, and the minimum distance between two clusters. It is able to find clusters in subspaces of varying size. The DOD measure is dependent on the threshold which is determined by the algorithm in an iterative manner [10]. The iterative phase that determines the best threshold adds significantly to the running time of the algorithm. Because of this, FINDIT employs sampling techniques like the other top-down algorithms. Sampling helps to improve performance, especially with very large datasets.

δ- CLUSTERS:

The existing cluster models may not always be adequate in capturing coherence exhibited among objects. Strong coherence may still exist among a set of objects even if they take quite different values on each attribute and the attribute values are not fully specified. This is very common in many applications including bio-informatics analysis as well as collaborative filtering analysis, where the data may be incomplete and subject to biases. A general model, referred as the cluster model is proposed to capture coherence exhibited by a subset of objects on a subset of attributes, while allowing absent attribute values. [7].

A δ-cluster essentially corresponds to a sub matrix that exhibits some coherent tendency. Formally, each δ-cluster can be uniquely identified by the set of relevant objects and attributes. A δ-cluster is similar to a subspace cluster in this respect, with the exception that the δ-cluster also allows missing values. Even though allowing missing values brings great flexibility to the δ-cluster model, the amount of missing entries in a δ-cluster should be limited to some extent to avoid trivial cases. The rule of the thumb is that, despite the missing values, there should still be sufficient evidence to demonstrate the coherency.

Clustering On Subsets of Attributes (COSA)

Clustering On Subsets of Attributes (COSA) is an iterative algorithm that assigns weights to each dimension for each instance, not each cluster. Starting with equally weighted dimensions, the algorithm examines the k nearest neighbors (knn) of each instance. These neighborhoods are used to calculate the respective dimension weights for each instance. Higher weights are assigned to those dimensions that have a smaller dispersion within the knn group. These weights are used to calculate dimension weights for pairs of instances which are in turn used to update the distances used in the knn calculation. The process is then repeated using the new distances until the weights stabilize. The neighborhoods for each instance become increasingly enriched with instances belonging to its own cluster. The dimension weights are refined as the dimensions relevant to a cluster receive larger weights [8]. The output is a distance matrix based on weighted inverse exponential distance and is suitable as input to any distance-based clustering method. After clustering, the weights of each dimension of cluster members are compared and an overall importance value for each dimension for each cluster is calculated. The number of dimensions in clusters need not be specified directly, but instead is controlled by a parameter, λ, that controls the strength of incentive for clustering on more dimensions. Each cluster may exist in different subspaces of different sizes, but they do tend to be of similar dimensionality. It also allows for the adjustment of the k used in the knn calculations. Friedman claims that the results are stable over a wide range of k values. The dimension weights are calculated for each instance and pair of instances, not for each cluster. After clustering the relevant dimensions must be calculated based on the dimension weights assigned to cluster members.

V. ANALYSIS

High dimensional data is increasingly common in many fields. The problem of curse of dimensionality has been studied extensively and there are various solutions, each appropriate for different types of high dimensional data and data mining procedures. Subspace clustering attempts to integrate feature evaluation and clustering in order to find clusters in different subspaces. Top-down algorithms simulate this integration by using multiple iterations of evaluation, selection, and clustering. This process selects a group of instances first and then evaluates the attributes in the context of that cluster of instances. This relatively slow approach combined with the fact that many are forced to use sampling techniques makes top-down algorithms more suitable for datasets with large clusters in relatively large subspaces. The clusters uncovered by top-down methods are often hyper-spherical in nature due to the use of cluster centers to represent groups of similar instances. The clusters form non-overlapping partitions of the dataset. Some algorithms allow for an additional group of outliers that contains instances not related to any cluster. Also, many require that the number of clusters and the size of the subspaces be input as parameters. The user must make use of domain knowledge to select and tune the input parameter settings. Bottom-up algorithms integrate the clustering and subspace selection by first selecting a subspace, then evaluating the instances in that context. This allows these algorithms to scale much more easily with both the number of instances in the dataset and the number of attributes. However, performance drops quickly with the size of the subspaces in which the clusters are found. The main parameter required by these algorithms is the density threshold. This can be difficult to set, especially across all dimensions of the dataset. Fortunately, even if some dimensions are mistakenly ignored due to improper thresholds, the algorithms may still find the clusters in a smaller subspace. Adaptive grid approaches help to alleviate this problem by allowing the number of bins in a dimension to change based on the characteristics of the data in that dimension. Often, bottom-up algorithms are able to find clusters of various shapes and sizes since the clusters are formed from various cells in a grid. This means that the clusters can overlap each other with one instance having the potential to be in more than one cluster. It is also possible for an instance to be considered an outlier and does not belong to any cluster. Clustering is a powerful data exploration tool capable of uncovering previously unknown patterns in data. Often, users have little knowledge of the data prior to clustering analysis and are seeking to find some interesting relationships to explore further. Unfortunately, all clustering algorithms require that the user set some parameters and make some assumptions about the clusters to be discovered.

VI. CONCLUSION

Subspace clustering algorithms allow users to break the assumption that all of the clusters in a dataset are found in the same set of dimensions. There are many potential applications with high dimensional data where subspace clustering approaches could help to uncover patterns missed by current clustering approaches. Applications in bioinformatics and text mining are particularly relevant and present unique challenges to subspace clustering. As with any clustering techniques, finding meaningful and useful results depends on the selection of the appropriate technique and proper tuning of the algorithm via the input parameters. Top-down algorithms simulate the integration by using multiple iterations of evaluation, selection, and clustering. This process selects a group of instances first and then evaluates the attributes in the context of that cluster of instances. Bottom-up algorithms integrate the clustering and subspace selection by first selecting a subspace, then evaluating the instances in the that context. However, each algorithm tries to mine efficient clustering results in its own style, but there is a high need for effective algorithms when dealing with high dimensional data.

REFERENCES

[1] A. Hinneburg and D. A. Keim, “Optimal grid clustering: Towards breaking the curse of dimensionality in high dimensional clustering,” in Proceedings of 25th International Conference on Very Large Data Bases, Edinburgh, Scotland, September, 1999, pp. 506-517.

[2] C. C. Aggarwal, J. L. Wolf, P. Yu, C. Procopiuc, and J. S. Park, “Fast algorithms for projected clustering,” in proceedings of the 1999 ACM SIGMOD International Conference on Management of Data, 1999, pp. 61-72.

[3] C.C. Aggarwal, P.S. Yu, “Finding generalized projected clusters in high dimensional spaces,” in proceedings of ACM SIGMOD International Conference on the Management of Data, 2000, pp. 70-81.

[4] C.C. Aggarwal, J.L. Wolf, P.S. Yu, et al., "Fast algorithms for projected clustering," in ACM SIGMOD Record, vol. 28:2, 1999, pp. 61-72.

[5] Goil, S., Nagesh, H. and Choudhary, A., "MAFIA: Efficient and scalable subspace clustering for very large data sets," Technical Report CPDC-TR-9906-010 Northwestern University, 1999.

[6] H. P. Kriegel, P. Kroger, M. Renz, and S. Wurst, "A generic framework for efficient subspace clustering of high dimensional data," in Proceedings of the 5th International Conference on Data Mining (ICDM), Houston, TX, 2005, pp. 250-257.

[7] J. Yang, W. Wang, H. Wang, and P. Yu, "δ-clusters:capturing subspace correlation in a large data set," in proceedings of 18th International Conference on Data Engineering, 2002, pp. 517-528.

[8] J. H. Friedman and J. J. Meulman, "Clustering objects on subsets of attributes," http://citeseer.nj.nec.com/friedman02clustering.html, 2002.

[9] K. Kailing, H.P. Kriegel and P. Kroger, "Density-connected subspace clustering for high dimensional data," in proceedings of the 4th SIAM International Conference on Data Mining, Orlando, FL, 2004, pp. 46-257.

[10] K.G. Woo, J.H. Lee, M.H. Kim, et al., "FINDIT: a fast and intelligent subspace clustering algorithm using dimension voting," Information and Software Technology 46(4) 2004, pp. 255-271.

[11] P. Lance, E. Haque, and H. Liu "Subspace clustering for high dimensional data: A review," in proceedings of ACM SIGKDD Explorations Newsletter, Vol. 6 Issue 1,2004, pp. 90–105.

[12] R. Agrawal, J. Gehrke, D. Gunopulos, P. Raghavan, "Automatic subspace clustering of high dimensional data for data mining applications," in Proceedings of the ACM SIGMOD International Conference on the Management of Data, 1998, pp. 94-105.

[13] Sunita Jahirabadkar, Parag Kulkarni "Clustering for High Dimensional Data: Density Based Subspace Clustering Algorithms," International Journal of Computer Applications (0975-8887) vol.63 :20, February 2013.

[14] Y. H. Chu, J. W. Huang, K. T. Chuang, D. N. Yang and M.S. Chen, "Density conscious subspace clustering for high dimensional data", IEEE Trans. Knowledge Data Eng. 22: 2010, pp. 16-30.

[15] Zhaohong Deng, Kup-Sze Choi, Yizhang Jiang, Jun Wang, Shitong Wang, "A Survey on Soft Subspace Clustering," Information Sciences: an International Journal, vol. 348: C, June 2016, pp. 84-106.